Schulinterner Lehrplan
zum Kernlehrplan für die gymnasiale Oberstufe

Informatik

(Stand: April 2019)

Inhalt

1 Die Fachgruppe Informatik im Werner-Heisenberg-Gymnasium

Am Werner-Heisenberg-Gymnasium werden in der Sekundarstufe I Inhalte, die das Fach Informatik betreffen, im Wahlpflichtbereich II (WP II) im Rahmen von fächerübergreifenden Kursen unterrichtet. In altersstufengerechter Weise wird hier strukturierendes Denken geschult, um die Welt „außerhalb des Computers“ besser zu verstehen. Dabei stehen der Blick in Anwendersysteme und die Modellierung realer Abläufe im Vordergrund.

Organisatorisch ist das Fach Informatik in der Sekundarstufe I in den MINT-Schwerpunkt der Schule eingebunden, den Schülerinnen und Schüler als Alternative zu einem sprachlichen Schwerpunkt anwählen können.

In der Sekundarstufe II bietet das Werner-Heisenberg-Gymnasium für die eigenen Schülerinnen und Schüler in allen Jahrgangsstufen jeweils einen Grundkurs in Informatik an.

Um insbesondere Schülerinnen und Schülern gerecht zu werden, die in der Sekundarstufe I keinen Informatikunterricht besucht haben, wird in Kursen der Einführungsphase besonderer Wert darauf gelegt, dass keine Vorkenntnisse aus der Sekundarstufe I zum erfolgreichen Durchlaufen des Kurses erforderlich sind.

Der Unterricht der Sekundarstufe II wird mit Hilfe der Programmiersprache Java durchgeführt.

Durch projektartiges Vorgehen, offene Aufgaben und Möglichkeiten, Problemlösungen zu verfeinern oder zu optimieren, entspricht der Informatikunterricht der Oberstufe in besonderem Maße den Erziehungszielen, Leistungsbereitschaft zu fördern, ohne zu überfordern.

Die stets kritische Entwicklung von Materialien und Unterrichtsvorhaben, die Evaluation von Lehr- und Lernprozessen sowie die regelmäßige Überprüfung und etwaige Modifikation des schulinternen Curriculums stellen einen wichtigen Beitrag zur Qualitätssicherung und -entwicklung des Unterrichts dar.

Ab dem Schuljahr 2019/20 stehen drei Computerräume mit jeweils 20 Computerarbeitsplätzen und einem digitalen Board sowie weitere Rechnerplätze in der Schülerbibliothek zur Verfügung. Zwei unmittelbar benachbarte dieser Räume lassen sich durch Aufschieben einer flexiblen Trennwand in einen großen Computer-saal verwandeln. Alle Arbeitsplätze sind an das schulinterne Rechnernetz angeschlossen, so dass Schülerinnen und Schüler über einen individuellen Zugang zum zentralen Server der Schule diese Arbeitsplätze zum Zugriff auf ihre eigenen Daten, zur Recherche im Internet oder zur Bearbeitung schulischer Aufgaben verwenden können.

2 Entscheidungen zum Unterricht

2.1 Unterrichtsvorhaben

Die Darstellung der Unterrichtsvorhaben im schulinternen Lehrplan besitzt den Anspruch, sämtliche im Kernlehrplan angeführten Kompetenzen abzudecken. Dies entspricht der Verpflichtung jeder Lehrkraft, Schülerinnen und Schülern Lerngelegenheiten zu ermöglichen, so dass alle Kompetenzerwartungen des Kernlehrplans von ihnen erfüllt werden können.

Die entsprechende Umsetzung erfolgt auf zwei Ebenen: der Übersichts- und der Konkretisierungsebene.

Im „Übersichtsraster Unterrichtsvorhaben“ (Kapitel 2.1.1) wird die für alle Lehrkräfte verbindliche Verteilung der Unterrichtsvorhaben dargestellt. Das Übersichtsraster dient dazu, einen schnellen Überblick über die Zuordnung der Unterrichtsvorhaben zu den einzelnen Jahrgangsstufen sowie den im Kernlehrplan genannten Kompetenzen, Inhaltsfeldern und inhaltlichen Schwerpunkten zu verschaffen. Der ausgewiesene Zeitbedarf versteht sich als grobe Orientierungsgröße, die nach Bedarf über- oder unterschritten werden kann.

Da in den folgenden Unterrichtsvorhaben Inhalte in der Regel anhand von Problemstellungen in Anwendungskontexten bearbeitet werden, werden in einigen Unterrichtsvorhaben jeweils mehrere Inhaltsfelder angesprochen.

2.1.1 Übersichtsraster Unterrichtsvorhaben

I) Einführungsphase

Einführungsphase	
Unterrichtsvorhaben E-I **Thema:** *Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten* **Zentrale Kompetenzen:** • Argumentieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Informatiksysteme • Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** • Einzelrechner • Dateisystem • Internet • Einsatz von Informatiksystemen **Zeitbedarf**: 2 Stunden	*Unterrichtsvorhaben E-II* **Thema**: *Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von einfachen Simulationen, Spielen* **Zentrale Kompetenzen:** • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Syntax und Semantik einer Programmiersprache **Zeitbedarf**: 11 Stunden

Einführungsphase	
Unterrichtsvorhaben E-III **Thema**: *Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Graphiken* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Syntax und Semantik einer Programmiersprache • Analyse, Entwurf und Implementierung einfacher Algorithmen **Zeitbedarf**: 19 Stunden	*Unterrichtsvorhaben E-IV* **Thema**: *Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von graphischen Spielen und graphischen Simulationen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Syntax und Semantik einer Programmiersprache • Analyse, Entwurf und Implementierung einfacher Algorithmen **Zeitbedarf**: 20 Stunden

<table>
<tr><th colspan="2">Einführungsphase</th></tr>
<tr><td><u>Unterrichtsvorhaben E-V</u>

Thema:
Such- und Sortieralgorithmen anhand kontextbezogener Beispiele

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Implementieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Algorithmen

Inhaltliche Schwerpunkte:
• Algorithmen zum Suchen und Sortieren
• Analyse, Entwurf und Implementierung einfacher Algorithmen
• Syntax und Semantik einer Programmiersprache

Zeitbedarf: 18 Stunden</td><td><u>Unterrichtsvorhaben E-VI</u>

Thema:
Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes

Zentrale Kompetenzen:
• Argumentieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Informatik, Mensch und Gesellschaft
• Informatiksysteme

Inhaltliche Schwerpunkte:
• Wirkungen der Automatisierung
• Geschichte der automatischen Datenverarbeitung
• Digitalisierung

Zeitbedarf: 6 Stunden</td></tr>
<tr><td colspan="2">Summe Einführungsphase: 76</td></tr>
</table>

II) Qualifikationsphase (Q1 und Q2) - GRUNDKURS

<table>
<tr><th colspan="2">Qualifikationsphase 1</th></tr>
<tr>
<td>

Unterrichtsvorhaben Q1-I

Thema:
Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung

Zentrale Kompetenzen:
- Argumentieren
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:
- Informatik, Mensch und Gesellschaft
- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten
- Informatiksysteme

Inhaltliche Schwerpunkte:
- Sicherheit
- Objekte und Klassen
- Analyse, Entwurf und Implementierung von Algorithmen
- Syntax und Semantik einer Programmiersprache
- Nutzung von Informatiksystemen

Zeitbedarf: 24 Stunden

</td>
<td>

Unterrichtsvorhaben Q1-II

Thema:
Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

Zentrale Kompetenzen:
- Argumentieren
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:
- Objekte und Klassen
- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache

Zeitbedarf: 14 Stunden

</td>
</tr>
</table>

Qualifikationsphase 1	
Unterrichtsvorhaben Q1-III **Thema:** *Suchen und Sortieren auf linearen Datenstrukturen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache **Zeitbedarf**: 10 Stunden	*Unterrichtsvorhaben Q1-IV* **Thema:** *Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** • Datenbanken • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache • Sicherheit **Zeitbedarf**: 23 Stunden

Qualifikationsphase 1	
Unterrichtsvorhaben Q1-V **Thema:** *Sicherheit und Datenschutz in Netzstrukturen* **Zentrale Kompetenzen:** • Argumentieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Informatiksysteme • Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** • Einzelrechner und Rechnernetzwerke • Sicherheit • Nutzung von Informatiksystemen, Wirkungen der Automatisierung **Zeitbedarf**: 18 Stunden	
Summe Qualifikationsphase 1: 89 Stunden	

Qualifikationsphase 2	
Unterrichtsvorhaben Q2-I **Thema:** *Modellierung und Implementierung von Anwendungen mit dynamischen nichtlinearen Datenstrukturen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache **Zeitbedarf**: 14 Stunden	*Unterrichtsvorhaben Q2-II* **Thema:** *Endliche Automaten und formale Sprachen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Endliche Automaten und formale Sprachen **Inhaltliche Schwerpunkte:** • Endliche Automaten • Grammatiken regulärer Sprachen • Möglichkeiten und Grenzen von Automaten und formalen Sprachen **Zeitbedarf**: 27 Stunden

<table>
<tr><th colspan="2">Qualifikationsphase 2</th></tr>
<tr><td><u>Unterrichtsvorhaben Q2-III</u>

Thema:
Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Zentrale Kompetenzen:
• Argumentieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Informatiksysteme
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Einzelrechner und Rechnernetzwerke
• Grenzen der Automatisierung

Zeitbedarf: 12 Stunden</td><td><u>Unterrichtsvorhaben Q2-IV</u>

Thema:
Wiederholung und Vertiefung ausgewählter Kompetenzen und zurückliegender Inhalte

Zeitbedarf: 12 Stunden</td></tr>
<tr><td colspan="2">Summe Qualifikationsphase 2: 65 Stunden</td></tr>
</table>

2.1.2 Konkretisierte Unterrichtsvorhaben

Im Folgenden sollen die im *Unterkapitel 2.1.1* aufgeführten Unterrichtsvorhaben konkretisiert werden.

In der Einführungs- und der Qualifikationsphase werden die integrierten Benutzeroberflächen BlueJ und/oder Greenfoot verwendet.

In der Qualifikationsphase werden die Unterrichtsvorhaben unter Berücksichtigung der Vorgaben für das Zentralabitur Informatik in NRW konkretisiert.

I) Einführungsphase

Die folgenden Kompetenzen aus dem Bereich *Kommunizieren und Kooperieren* werden in allen Unterrichtsvorhaben der Einführungsphase vertieft und sollen aus Gründen der Lesbarkeit nicht in jedem Unterrichtsvorhaben separat aufgeführt werden:

Die Schülerinnen und Schüler

- verwenden Fachausdrücke bei der Kommunikation über informatische Sachverhalte (K),
- präsentieren Arbeitsabläufe und -ergebnisse (K),
- kommunizieren und kooperieren in Gruppen und in Partnerarbeit (K),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K).

Unterrichtsvorhaben EF-I

Thema: Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten

Leitfragen: *Womit beschäftigt sich die Wissenschaft der Informatik? Wie kann die in der Schule vorhandene informatische Ausstattung genutzt werden?*

Vorhabenbezogene Konkretisierung:
Das erste Unterrichtsvorhaben stellt eine allgemeine Einführung in das Fach Informatik dar. Dabei ist zu berücksichtigen, dass für manche Schülerinnen und Schüler in der Einführungsphase der erste Kontakt mit dem Unterrichtsfach Informatik stattfindet, sodass zu Beginn Grundlagen des Fachs behandelt werden müssen.

Zunächst wird auf den Begriff der Information eingegangen und die Möglichkeit der Kodierung in Form von Daten thematisiert. Anschließend wird auf die Übertragung von Daten im Sinne des Sender-Empfänger-Modells eingegangen. Dabei wird eine überblickartige Vorstellung der Kommunikation von Rechnern in Netzwerken erarbeitet.

Des Weiteren soll der grundlegende Aufbau eines Rechnersystems im Sinne der Von-Neumann-Architektur erarbeitet werden und mit dem grundlegenden Prinzip der Datenverarbeitung (Eingabe-Verarbeitung-Ausgabe) in Beziehung gesetzt werden.

Bei der Beschäftigung mit Datenkodierung, Datenübermittlung und Datenverarbeitung ist jeweils ein Bezug zur konkreten Nutzung der informatischen Ausstattung der Schule herzustellen. So wird in die verantwortungsvolle Nutzung dieser Systeme eingeführt.

Zeitbedarf: 2 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Information, deren Kodierung und Speicherung (a) Informatik als Wissenschaft der Verarbeitung von Informationen (b) Darstellung von Informationen in Schrift, Bild und Ton (c) Speichern von Daten mit informatischen Systemen am Beispiel der Schulrechner (d) Vereinbarung von Richtlinien zur Datenspeicherung auf den Schulrechnern (z.B. Ordnerstruktur, Dateibezeichner usw.)	Die Schülerinnen und Schüler • beschreiben und erläutern den Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A), • nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D), • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).	*Beispiel*: Textkodierung Kodierung und Dekodierung von Texten mit unbekannten Zeichensätzen (z.B. Wingdings) *Beispiel*: Bildkodierung Kodierung von Bildinformationen in Raster- und Vektorgrafiken
2. Informations- und Datenübermittlung in Netzen (a) „Sender-Empfänger-Modell“ und seine Bedeutung für die Eindeutigkeit von Kommunikation (b) Informatische Kommunikation in Rechnernetzen am Beispiel des Schulnetzwerks (z.B. Benutzeranmeldung, Netzwerkordner, Zugriffsrechte, Client-Server) (c) Grundlagen der technischen Umsetzung von Rechnerkommunikation am Beispiel des Internets (z.B. Netzwerkadresse, Paketvermittlung, Protokoll)		*Beispiel*: Rollenspiel zur Paketvermittlung im Internet Schülerinnen und Schüler übernehmen die Rollen von Clients und Routern. Sie schicken spielerisch Informationen auf Karten von einem Schüler-Client zum anderen. Jede Schülerin und jeder Schüler hat eine Adresse, jeder Router darüber hinaus eine Routingtabelle. Mit Hilfe der Tabelle und einem Würfel wird entschieden, wie ein Paket weiter vermittelt wird.

(d) Richtlinien zum verantwortungsvollen Umgang mit dem Internet		
3. Aufbau informatischer Systeme (a) Identifikation typischer Komponenten informatischer Systeme und anschließende Beschränkung auf das Wesentliche, Herleitung der „Von-Neumann-Architektur“ (b) Identifikation des EVA-Prinzips (Eingabe-Verarbeitung-Ausgabe) als Prinzip der Verarbeitung von Daten und Grundlage der „Von-Neumann-Architektur“		mögliches Vorgehen: Internetrecherche

Unterrichtsvorhaben EF-II

Thema: Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von einfachen Simulationen, Spielen

Leitfrage: *Wie lassen sich Gegenstandsbereiche informatisch modellieren und im Sinne einer Simulation informatisch realisieren?*

Vorhabenbezogene Konkretisierung:
Ein zentraler Bestandteil des Informatikunterrichts der Einführungsphase ist die objektorientierte Programmierung. Dieses Unterrichtsvorhaben führt in die Grundlagen der Analyse, Modellierung und Implementierung in diesem Kontext ein.

Dazu werden zunächst konkrete Gegenstandsbereiche aus der Lebenswelt der Schülerinnen und Schüler analysiert und im Sinne des objektorientierten Paradigmas strukturiert. Dabei werden die grundlegenden Begriffe der Objektorientierung und Modellierungswerkzeuge wie Objektkarten, Klassenkarten oder Beziehungsdiagramme eingeführt.

Im Anschluss wird mit der Realisierung erster Projekte begonnen. Die teils vorgegebenen, teils selbst zu entwickelnden Klassen werden von Schülerinnen und Schülern in Teilen analysiert und entsprechende Objekte anhand einfacher Problemstellungen erprobt. Dazu muss der grundlegende Aufbau einer Java-Klasse thematisiert und zwischen Deklaration, Initialisierung und Methodenaufrufen unterschieden werden.

Da bei der Umsetzung dieser ersten Projekte konsequent auf die Verwendung von Kontrollstrukturen verzichtet wird und der Quellcode aus einer rein linearen Sequenz besteht, ist auf diese Weise eine Fokussierung auf die Grundlagen der Objektorientierung möglich, ohne dass algorithmische Probleme ablenken. Natürlich kann die Arbeit an diesen Projekten unmittelbar zum nächsten Unterrichtsvorhaben führen. Dort stehen unter anderem Kontrollstrukturen im Mittelpunkt.

Zeitbedarf: 11 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Identifikation von Objekten (a) An einem lebensweltnahen Beispiel werden Objekte im Sinne der objektorientierten Modellierung eingeführt. (b) Objekte werden mit Objektkarten visualisiert und mit sinnvollen Attributen und „Fähigkeiten“, d.h. Methoden versehen. (c) Manche Objekte sind prinzipiell typgleich und werden so zu einer Objektsorte bzw. Objektklasse zusammengefasst. (d) Klassendiagramme, Vertiefung: Modellierung weiterer Beispiele	Die Schülerinnen und Schüler • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M), • stellen die Kommunikation zwischen Objekten grafisch dar (M), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I), • stellen den Zustand eines Objekts dar (D).	*Beispiele:* Auto, Aufzug-Kabine
2. Analyse von Klassen (a) Objektorientierte Programmierung als modularisiertes Vorgehen (b) Teilanalyse des Aufbaus von Klassen		*Beispiel:* Krümelmonster
3. Implementierung (a) Grundaufbau einer Java-Klasse (b) Konstruktoren (c) Deklaration, Erzeugung und Initialisierung von Objekten (d) Methodenaufrufe mit Parameterübergabe zur Manipulation von Objekteigenschaften		*Beispiele:* Auto, Aufzug-Kabine, Krümelmonster

Unterrichtsvorhaben EF-III

Thema: Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Graphiken

Leitfragen: *Wie lassen sich einfache Graphiken unter der Steuerung durch Benutzereingaben realisieren?*

Vorhabenbezogene Konkretisierung:
Der Schwerpunkt dieses Unterrichtsvorhabens liegt auf der Entwicklung mehrerer Projekte, die durch Eingaben des Benutzers gesteuerte Graphiken aufweisen. Für die Projektumsetzung werden Kontrollstrukturen in Form von Schleifen und Verzweigungen benötigt und eingeführt.

Das Unterrichtsvorhaben umfasst auch die Erstellung und Nutzung graphischer Benutzeroberflächen (GUI). Projektelemente müssen zu sinnhaften eigenen Klassen zusammengefasst werden, die dann ihre eigenen Attribute und Dienste besitzen.

Komplexere Assoziationsbeziehungen zwischen Klassen werden in diesem Unterrichtsvorhaben zunächst nicht behandelt. Sie stellen den Schwerpunkt des folgenden Vorhabens dar.

Zeitbedarf: 19 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Berechnungen in Implementationen (a) Mehrstufige Berechnungen (b) Verwendung lokaler Variablen (c) Kommunikation der und über Methodenaufrufe (d) Reflexion über Objektorientierung (e) Exceptions als Kontrollstruktur **2. Kommunikation zwischen Programm und Benutzer** (a) Paket-Importe (b) Nutzung von Klassenbibliotheken (c) Arbeiten mit GUI, auch unter Hinzunahme graphischer Darstellungen **3. Kombination mehrerer Graphiken als Beispiel ausgebauter Assoziationsbeziehungen zwischen Klassen** (a) Mögliche Vertiefung: Interfaces am Beispiel der Sichten (b) Verzweigungen	Die Schülerinnen und Schüler • analysieren und erläutern einfache Algorithmen und Programme (A), • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M), • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M), • modifizieren einfache Algorithmen und Programme (I), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I), • testen Programme schrittweise anhand von Beispielen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	*Beispiele:* Bruchprojekt, Quaderberechnung

Unterrichtsvorhaben EF-IV

Thema: Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen und grafischen Simulationen

Leitfrage: *Wie lassen sich komplexere Datenflüsse und Beziehungen zwischen Objekten und Klassen realisieren?*

Vorhabenbezogene Konkretisierung:
Dieses Unterrichtsvorhaben beschäftigt sich im Schwerpunkt mit dem Aufbau komplexerer Objektbeziehungen. Während in vorangegangenen Unterrichtsvorhaben Objekte nur jeweils solchen Objekten Nachrichten schicken konnten, die sie selbst erstellt haben, soll in diesem Unterrichtsvorhaben diese hierarchische Struktur aufgebrochen werden.

Dazu bedarf es zunächst einer präzisen Unterscheidung zwischen Objektreferenzen und Objekten, so dass klar wird, dass Dienste eines Objektes von unterschiedlichen Objekten über unterschiedliche Referenzen in Anspruch genommen werden können. Auch der Aufbau solcher Objektbeziehungen muss thematisiert werden. Des Weiteren wird das Prinzip der Vererbung im objektorientierten Sinne angesprochen. Dazu werden die wichtigsten Varianten der Vererbung anhand von verschiedenen Projekten vorgestellt. Zunächst wird die Vererbung als Spezialisierung im Sinne einer einfachen Erweiterung einer Oberklasse vorgestellt. Darauf folgt ein Projekt, welches das Verständnis von Vererbung um den Aspekt der späten Bindung erweitert, indem Dienste einer Oberklasse überschrieben werden. Modellierungen sollen in Form von Implementationsdiagrammen erstellt werden.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Spezifizierungen zu Klassenbeziehungen (a) Vererbung (b) Geheimnisprinzip (c) die Klasse *Object* (d) der Cast-Operator (e) Überschreiben (f) Überladen	Die Schülerinnen und Schüler • analysieren und erläutern eine objektorientierte Modellierung (A), • stellen die Kommunikation zwischen Objekten grafisch dar (M), • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M), • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M), • modellieren Klassen unter Verwendung von Vererbung (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • testen Programme schrittweise anhand von Beispielen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • modifizieren einfache Algorithmen und Programme (I), • stellen Klassen, Assoziations- und Verer-	
2. Entwicklung von Spielen (a) Modellierung eines Spiels (b) Dokumentation der Klassen des Projekts (c) Analyse (d) Implementierung (e) zusätzliche Assoziationsbeziehungen in Diagramm, Dokumentation und Quellcode (f) Verallgemeinerung der neuen Verwendung von Objektreferenzen und Reflexion		*Beispiele:* Schatzräuber, Kamelrennen, Jäger-Beute-Simulation
3. Diagramme und Assoziationen (a) Entwurfsdiagramme (b) Implementationsdiagramm und Quellcode (c) Multiplizitäten (d) Aggregation		*Software:* UMLed

(e) Komposition	bungsbeziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D).	
4. Schleifen (a) Gegenüberstellung verschiedener Schleifentypen (b) einfache Zeichnungen als Anwendung (c) iterierte Zeichnungen als Anwendung (d) eindimensionale Datenfelder als Kontroll- oder Speicherstruktur		*Hilfsmittel:* turtle von Alfred Hermes *Beispiel:* Projekt *Zahlenliste*

Unterrichtsvorhaben EF-V

Thema: Such- und Sortieralgorithmen anhand kontextbezogener Beispiele

Leitfragen: *Wie können Objekte bzw. Daten effizient sortiert werden, sodass eine schnelle Suche möglich wird?*

Vorhabenbezogene Konkretisierung:
Dieses Unterrichtsvorhaben beschäftigt sich mit der Erarbeitung von Such- und Sortieralgorithmen.

Zunächst erarbeiten die Schülerinnen und Schüler mögliche Einsatzszenarien für Such- und Sortieralgorithmen, um sich der Bedeutung einer effizienten Lösung dieser Probleme bewusst zu werden. Anschließend werden Strategien zur Sortierung erarbeitet und hinsichtlich der Anzahl notwendiger Vergleiche auf ihre Effizienz untersucht.

Die erarbeiteten Strategien werden systematisiert und implementiert.

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Explorative Erarbeitung eines Sortierverfahrens (a) Sortierprobleme im Kontext informatischer Systeme und im Alltag (z.B. Dateisortierung, Tabellenkalkulation, Telefonbuch, Bundesligatabelle usw.) (b) Vergleich zweier Elemente als Grundlage eines Sortieralgorithmus (c) Erarbeitung eines Sortieralgorithmus durch die Schülerinnen und Schüler	Die Schülerinnen und Schüler • beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A), • entwerfen einen weiteren Algorithmus zum Sortieren (M), • analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D), • testen Programme schrittweise anhand von Beispielen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • modifizieren einfache Algorithmen und Programme (I), • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I).	
2. Implementation, Systematisierung von Algorithmen und Effizienzbetrachtungen (a) Formulierung (falls selbst gefunden) oder Erläuterung von mehreren Algorithmen (b) Anwendung von Sortieralgorithmen auf verschiedene Beispiele (c) Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche (d) Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs		*Beispiele:* direktes Auswerten, bubbleSort, direktes Einfügen, bucketSort, randomSort, bogoSort, heapSort
3. Sortieren im Kontext (a) Textarbeit (b) ggf. Vertiefung / Anwendung		*Beispiele:* Arbeiten mit Texten, Wort-Spiele (Wortsalat, Hangman), Objektvergleiche, Schreiben und Lesen von Daten

Unterrichtsvorhaben EF-VI

Thema: Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes

Leitfrage: *Welche Entwicklung durchlief die moderne Datenverarbeitung und welche Auswirkungen ergeben sich insbesondere hinsichtlich neuer Anforderungen an den Datenschutz daraus?*

Vorhabenbezogene Konkretisierung:
Das folgende Unterrichtsvorhaben stellt den Abschluss der Einführungsphase dar. Schülerinnen und Schüler sollen selbstständig informatische Themenbereiche aus dem Kontext der Geschichte der Datenverarbeitung und insbesondere den daraus sich ergebenen Fragen des Datenschutzes bearbeiten. Diese Themenbereiche können in Kleingruppen bearbeitet und in Form von Plakatpräsentationen vorgestellt werden. Schülerinnen und Schüler sollen dabei mit Unterstützung der Lehrperson selbstständige Recherchen zu ihren Themen anstellen und auch eine sinnvolle Eingrenzung ihres Themas vornehmen.

Anschließend wird verstärkt auf den Aspekt des Datenschutzes eingegangen. Dazu wird das Bundesdatenschutzgesetz oder die europäische Datenschutzgrundverordnung in Auszügen behandelt und auf schülernahe Beispielsituationen zur Anwendung gebracht. Dabei steht keine formale juristische Bewertung der Beispielsituationen im Vordergrund, die im Rahmen eines Informatikunterrichts auch nicht geleistet werden kann, sondern vielmehr eine persönliche Einschätzung von Fällen im Geiste des Datenschutzgesetzes.

Zeitbedarf: 6 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Selbstständige Erarbeitung von Themen durch die Schülerinnen und Schüler (a) Mögliche Themen zur Erarbeitung in Kleingruppen: • „Eine kleine Geschichte der Digitalisierung: vom Morsen zum modernen Digitalcomputer“ • „Eine kleine Geschichte der Kryptographie: von Caesar zur Enigma“ • „Von Nullen, Einsen und mehr: Stellenwertsysteme und wie man mit ihnen rechnet“ • „Kodieren von Texten und Bildern: ASCII, RGB und mehr“ • „Auswirkungen der Digitalisierung: Veränderungen der Arbeitswelt und Datenschutz“ (b) Vorstellung und Diskussion durch Schülerinnen und Schüler	Die Schülerinnen und Schüler • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A), • erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A), • stellen ganze Zahlen und Zeichen in Binärcodes dar (D), • interpretieren Binärcodes als Zahlen und Zeichen (D), • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation. (K).	*Beispiel:* Ausstellung zu informatischen Themen Die Schülerinnen und Schüler bereiten eine Ausstellung zu informatischen Themen vor. Dazu werden Stellwände und Plakate vorbereitet. *Materialien:* Schülerinnen und Schüler recherchieren selbstständig im Internet, in der Schulbibliothek, in öffentlichen Bibliotheken usw.
2. Vertiefung des Themas Datenschutz (a) Erarbeitung grundlegender Begriffe und Prinzipien des Datenschutzes (b) Problematisierung und Anknüpfung an die Lebenswelt der Schülerinnen und Schüler (c) Diskussion und Bewertung von Fallbeispielen aus dem Themenbereich „Datenschutz“		*Beispiel:* Fallbeispiele aus dem aktuellen Tagesgeschehen Die Schülerinnen und Schüler bearbeiten Fallbeispiele aus ihrer eigenen Erfahrungswelt oder der aktuellen Medienberichterstattung. *Materialien:* Materialblatt zum Bundesdatenschutzgesetz

II) Qualifikationsphase

Die folgenden Kompetenzen aus dem Bereich *Kommunizieren und Kooperieren* werden in allen Unterrichtsvorhaben der Qualifikationsphase vertieft und sollen aus Gründen der Lesbarkeit nicht in jedem Unterrichtsvorhaben separat aufgeführt werden:

Die Schülerinnen und Schüler

- verwenden die Fachsprache bei der Kommunikation über informatische Sachverhalte (K),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung von Dateien unter Berücksichtigung der Rechteverwaltung (K),
- organisieren und koordinieren kooperatives und eigenverantwortliches Arbeiten (K),
- strukturieren den Arbeitsprozess, vereinbaren Schnittstellen und führen Ergebnisse zusammen (K),
- beurteilen Arbeitsorganisation, Arbeitsabläufe und Ergebnisse (K),
- präsentieren Arbeitsabläufe und -ergebnisse adressatengerecht (K).

Unterrichtsvorhaben Q1-I:

Thema: Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung

Leitfragen: *Wie modelliert und implementiert man zu einer Problemstellung in einem geeigneten Anwendungskontext Java-Klassen inklusive ihrer Attribute, Methoden und Beziehungen? Wie kann man die Modellierung und die Funktionsweise der Anwendung grafisch darstellen? Worauf sollte man im Sinne des Datenschutzes und der gültigen Urheberrechtslizenzen bei der Benutzung fremder Bestandteile für eigene Projekte achten?*

Vorhabenbezogenen Konkretisierung:
Zu einer Problemstellung in einem Anwendungskontext soll eine Java-Anwendung entwickelt werden. Die Problemstellung soll so gewählt sein, dass für diese Anwendung die Verwendung einer abstrakten Oberklasse sowie rekursive Methoden sinnvoll erscheinen. Um die Aufgabe einzugrenzen, können (nach der ersten Problemanalyse) einige Teile (Modellierungen oder Teile von Java-Klassen) vorgegeben werden. In Abgrenzung vom nächsten Unterrichtsvorhaben und als inhaltliche Vorbereitung auf dynamische Datenstrukturen wird bei diesem Anwendungskontext ein mehrdimensionales Datenfeld benutzt.

Die Schülerinnen und Schülern machen sich zunächst mit dem Phänomen der Rekursion in der Informatik vertraut und erläutern und modifizieren sodann den ersten Entwurf und modellieren sowie implementieren weitere Klassen und Methoden für eine entsprechende Anwendung. Klassen und ihre Beziehungen können in einem Implementationsdiagramm dargestellt werden. Dabei können Sichtbarkeitsbereiche zugeordnet werden. Exemplarisch bietet sich die Dokumentation einer Klasse an. Der Nachrichtenaustausch zwischen verschiedenen Objekten wird verdeutlicht. In diesem Zusammenhang wird das Nachrichtenkonzept der objektorientierten Programmierung wiederholt.

Inwieweit von Fremden programmierte oder bereitgestellte Materialien für die eigene Projektentwicklung verwendet werden darf, wird im Rahmen einer Wiederholung der Grundlagen des Datenschutzes sowie einer Einführung in bestehende Urheberrechtslizenzen thematisiert.

Zeitbedarf: 24 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Datenschutz und Urheberrechtslizenzen – Wiederholung objektorientierter Programmierung (a) Grundlagen des Datenschutzes (Wiederholung) (b) Beachten und eigenes Setzen von *creative commons* (c) Quelltextanalyse und –interpretation zu einem vorgegebenen Projekt (d) eigenständige Implementation zu einem Ratespiel	Die Schülerinnen und Schüler • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A), • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementie-	*Beispiele*: (a) Wortsalat (b) Zahlenraten

	rung und zum Test von Informatiksystemen an (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	
2. Mathematische, grafische und algorithmische Rekursionen (a) mathematische Rekursionen (b) grafische Rekursionen in Java (c) rekursive Algorithmen	Die Schülerinnen und Schüler • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • analysieren und erläutern objektorientierte Modellierungen (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • ordnen Klassen, Attributen und Me-	*Beispiele*: (a) Fakultät, Fibonaccizahlen (b) Kochkurve, Sirpinski-Dreieck, Schaf-garbe, Pythagoras-Baum (c) mythologisches Labyrinth (Theseus)

	thoden ihre Sichtbarkeitsbereiche zu (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	
3. optionale Vertiefung: Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels (a) Analyse der Problemstellung (b) Analyse der Modellierung (Implementationsdiagramm) (c) Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse) (d) Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung) (e) Dokumentation von Klassen (f) Implementierung der Anwendung	Die Schülerinnen und Schüler • analysieren und erläutern objektorientierte Modellierungen (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von	*Beispiel:* Minesweeper

oder von Teilen der Anwendung	Vererbung durch Spezialisieren und Generalisieren (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen (D), • stellen die Kommunikation zwischen Objekten grafisch dar (D).	

Unterrichtsvorhaben Q1-II:

Thema: Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

Leitfrage: *Wie können beliebig viele linear angeordnete Daten im Anwendungskontext verwaltet werden?*

Vorhabenbezogene Konkretisierung:
Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext, in dem Daten nach dem First-In-First-Out-Prinzip verwaltet werden, werden der Aufbau von Schlangen dargestellt und die Operationen der Klasse Queue erläutert. Anschließend werden für die Anwendung notwendige Klassen modelliert und implementiert. Ferner wird die Anwendung modifiziert, um den Umgang mit der Datenstruktur zu üben. Nach Erläuterungen, wie Daten nach dem Last-In-First-Out-Prinzip verwaltet werden, werden Unterschiede zwischen den Datenstrukturen Schlange und Stapel erarbeitet. Um einfacher an Objekte zu gelangen, die zwischen anderen gespeichert sind, wird die Klasse List eingeführt. In mindestens einem weiteren Anwendungskontext wird die Verwaltung von Daten in Schlangen, Stapeln oder Listen vertieft. Modellierungen werden dabei in Entwurfs- und Implementationsdiagrammen dargestellt.

Zeitbedarf: 14 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Die Datenstruktur Schlange (a) Dokumentation der Klasse Queue (b) Erarbeitung der Funktionalität der Klasse Queue (c) Modellierung und Implementierung der Klasse Queue (d) Anwendungskontext (e) Zählalgorithmus	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nicht-	*Beispiel:* Druckerwarteschlange
2. Die Datenstruktur Stapel (a) Dokumentation der Klasse Stack (b) Erarbeitung der Funktionalität der Klasse Stack (c) Modellierung und Implementierung der Klasse Stack		*Beispiel:* Korrektursimulation

(d) Zählalgorithmus (e) Anwendungskontext	lineare Datensammlungen zu (M), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modifizieren Algorithmen und Programme (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D).	
3. Die Datenstruktur lineare Liste (a) Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen (b) Dokumentation der Klasse List (c) Erarbeitung der Funktionalität der Klasse List (d) ggf. Modellierung und Implementierung der Klasse List (e) Zählalgorithmus		
4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext		*Beispiel:* Prioritätenwarteschlange, zum Beispiel auch als Arztpraxisprojekt

Unterrichtsvorhaben Q1-III:

Thema: Suchen und Sortieren auf linearen Datenstrukturen

Leitfrage: *Wie kann man gespeicherte Informationen günstig (wieder-)finden?*

Vorhabenbezogene Konkretisierung:
Informationen in einer linearen Liste bzw. einem Feld werden gesucht. Hierzu werden Verfahren entwickelt und implementiert bzw. analysiert und erläutert, wobei neben einem iterativen auch ein rekursives Verfahren thematisiert wird und mindestens ein Verfahren selbst entwickelt und implementiert wird. Die verschiedenen Verfahren werden hinsichtlich Speicherbedarf und Zahl der Vergleichsoperationen miteinander verglichen.

Anschließend werden Sortierverfahren entwickelt und implementiert (ebenfalls für lineare Listen und Felder). Hierbei soll auch ein rekursives Sortierverfahren entwickelt werden. Die Implementationen aus der Einführungsphase werden wiedererkannt.

Abschließend werden verschiedene Sortierverfahren hinsichtlich der Anzahl der benötigten Vergleichsoperationen und des Speicherbedarfs beurteilt.

Zeitbedarf: 10 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien

1. Suchen von Daten in Listen und Arrays	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	
2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren		*Beispiele:* rekursive Sortierverfahren wie quickSort oder mergeSort, Wiederholung der Sortierverfahren aus der Einführungsphase, Anwendung etwa in Verwaltungsbeispielen
3. Untersuchung der Effizienz von Sortierverfahren		

Unterrichtsvorhaben Q1-IV:

Thema: Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Leitfragen*: Wie können Fragestellungen mit Hilfe einer Datenbank beantwortet werden? Wie entwickelt man selbst eine Datenbank für einen Anwendungskontext?*

Vorhabenbezogene Konkretisierung:
Ausgehend von einer vorhandenen Datenbank erfahren Schülerinnen und Schüler relevante Fragestellungen, die mit dem vorhandenen Datenbestand beantwortet werden sollen. Zur Beantwortung dieser Fragestellungen wird die vorgegebene Datenbank von den Schülerinnen und Schülern analysiert, und die notwendigen Grundbegriffe für Datenbanksysteme sowie die erforderlichen SQL-Abfragen werden erarbeitet.

In anderen Anwendungskontexten müssen Datenbanken erst noch entwickelt werden, um Daten zu speichern und Informationen für die Beantwortung von möglicherweise auftretenden Fragen zur Verfügung zu stellen. Dafür ermitteln Schülerinnen und Schüler in den Anwendungssituationen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten und stellen diese in Entity-Relationship-Modellen dar. Entity-Relationship-Modelle werden interpretiert und erläutert, modifiziert und in Datenbankschemata überführt. Mit Hilfe von SQL-Anweisungen können anschließend im Kontext relevante Informationen aus der Datenbank extrahiert werden.

Ein Entity-Relationship-Diagramm kann auch verwendet werden, um die Entitäten inklusive ihrer Attribute und Relationen in einem vorgegebenen Datenbankschema darzustellen.

Es wird verdeutlicht, dass in Datenbanken Redundanzen unerwünscht sind und Konsistenz gewährleistet sein sollte. Die 1. bis 3. Normalform wird als Gütekriterium für Datenbankentwürfe eingeführt. Datenbankschemata werden hinsichtlich der 1. bis 3. Normalform untersucht und (soweit nötig) normalisiert.

Abschließend werden Datenschutzfragen im Umgang mit Datenbanken thematisiert.

Zeitbedarf: 23 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken (a) Aufbau von Datenbanken und Grundbegriffe • Fragestellungen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema (b) SQL-Abfragen • Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) …FROM, WHERE, AND, OR, NOT) auf einer Tabelle • Analyse und Erarbeitung von SQL-Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (…), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL) (c) Vertiefung	Die Schülerinnen und Schüler • erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A), • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A), • analysieren und erläutern eine Datenbankmodellierung (A), • erläutern die Eigenschaften normalisierter Datenbankschemata (A), • bestimmen Primär- und Sekundärschlüssel (M), • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), • modifizieren eine Datenbankmodellierung (M), • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), • bestimmen Primär- und Sekundärschlüssel (M), • überführen Datenbankschemata in vorgegebene Normalformen (M), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I), • ermitteln Ergebnisse von Datenbankabfra-	*Beispiel:* Fitnesscenter
2. Modellierung von relationalen Datenbanken (a) Entity-Relationship-Diagramm • Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und		*Beispiel:* Fahrradverleih Der Fahrradverleih *BTR (BikesToRent)* verleiht unterschiedliche Typen von Fahrrädern diverser Firmen an seine Kunden. Die Kunden sind bei *BTR* registriert (Name, Adresse, Telefon). *BTR* kennt von den Fahrradfirmen den Namen und die

Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms • Erläuterung und Modifizierung einer Datenbankmodellierung (b) Entwicklung einer Datenbank aus einem Datenbankentwurf • Modellierung eines relationalen Datenbankschematas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln (c) Redundanz, Konsistenz und Normalformen • Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation • Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)	gen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).	Telefonnummer. Kunden von *BTR* können CityBikes, Treckingräder und Mountainbikes ausleihen. *Beispiel*: Reederei Die Datenverwaltung einer Reederei soll in einem Datenbanksystem umgesetzt werden. Ausgehend von der Modellierung soll mit Hilfe eines ER-Modells und eines Datenbankschemas dieser erste Entwurf normalisiert und in einem Datenbanksystem umgesetzt werden. Es schließen sich diverse SQL-Abfragen an, wobei auf die Relationenalgebra eingegangen wird. *Beispiel:* Buchungssystem In dem Online-Buchungssystem einer Schule können die Lehrer Medienräume, Beamer, Laptops, Kameras, usw. für einen bestimmten Zeitpunkt buchen, der durch Datum und die Schulstunde festgelegt ist. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden. Unter http://mrbs.sourceforge.net (abgerufen: 30.03. 2014) findet man ein freies Online-Buchungssystem inklusive Demo, an Hand derer man erläutern kann, worum es in dem Projekt geht. *Beispiel: Schulverwaltung* In einer Software werden die Schulhalbjahre, Jahrgangsstufen, Kurse, Klassen, Schüler, Lehrer und Noten einer Schule verwaltet. Man kann dann ablesen, dass z.B. Schüler X von Lehrer Y im x. Halbjahr des Schuljahrs y in der Jahrgangsstufe 9 im Differenzierungsbereich im Fach Infor-

		matik die Note „sehr gut“ erhalten hat. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden und das Thema Datenschutz besprochen werden.
3. Datenschutzfragen im Umgang mit Daternbanken		*Beispiel:* elektronisches Klassenbuch

Unterrichtsvorhaben Q1-V:

Thema: Sicherheit und Datenschutz in Netzstrukturen

Leitfragen: *Wie werden Daten in Netzwerken übermittelt? Was sollte man in Bezug auf die Sicherheit beachten?*

Vorhabenbezogene Konkretisierung:
Anschließend an das vorhergehende Unterrichtsvorhaben zum Thema Datenbanken werden der Datenbankzugriff aus dem Netz, Topologien von Netzwerken, eine Client-Server-Struktur, das TCP/IP-Schichtenmodell sowie Sicherheitsaspekte beim Zugriff auf Datenbanken und verschiedene symmetrische und asymmetrische kryptografische Verfahren analysiert und erläutert.

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken (a) Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs (b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz (c) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-	Die Schülerinnen und Schüler • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A), • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A),	*Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken (Download Q1-V.1)

Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen	• untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D).	

Unterrichtsvorhaben Q2-I:

Thema: Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen

Leitfragen: *Wie können Daten im Anwendungskontext mit Hilfe binärer Baumstrukturen verwaltet werden? Wie kann dabei der rekursive Aufbau der Baumstruktur genutzt werden? Welche Vor- und Nachteile haben Suchbäume für die geordnete Verwaltung von Daten?*

Vorhabenbezogene Konkretisierung:
Für Baumstrukturen werden grundlegende Begriffe eingeführt und der rekursive Aufbau binärer Bäume dargestellt.
Anschließend werden Klassen modelliert und implementiert. Dabei werden die Operationen der Datenstruktur Binärbaum thematisiert und die entsprechende Klasse BinaryTree (der Materialien für das Zentralabitur in NRW) verwendet. Klassen und ihre Beziehungen werden in Entwurfs- und Implementationsdiagrammen dargestellt. Die Funktionsweise von Methoden wird anhand grafischer Darstellungen von Binärbäumen erläutert.

Unter anderem sollen die verschiedenen Baumtraversierungen (Pre-, Post- und Inorder-) implementiert werden. Unterschiede bezüglich der Möglichkeit, den Baum anhand der Ausgabe der Bauminhalte via Pre-, In- oder Postorder-Traversierung zu rekonstruieren, werden dabei ebenfalls angesprochen, indem die fehlende Umkehrbarkeit der Zuordnung Binärbaum → Inorder-Ausgabe an einem Beispiel verdeutlicht wird.

Eine Tiefensuche wird verwendet, um einen in der Baumstruktur gespeicherten Inhalt zu suchen.

Die Operationen der Datenstruktur Suchbaum werden thematisiert und unter der Verwendung der Klasse BinarySearchTree (der Materialien für das Zentralabitur in NRW) weitere Klassen oder Methoden in diesem Anwendungskontext modelliert und implementiert. Auch in diesem Kontext werden grafische Darstellungen der Bäume verwendet.

Zeitbedarf: 14 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Analyse von Baumstrukturen (a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit) (b) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • modifizieren Algorithmen und Programme	*Beispiel:* Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht. oder *Beispiel*: Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat. *Weitere Beispiele für Anwendungskontexte für binäre Bäume:* *Beispiel*: Suchbäume (zur sortierten Speicherung von Daten) Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle, die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) *oder* *Beispiel*: Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum. *oder*

	(I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	*Beispiel*: Codierungsbäume für Codierungen, deren Alphabet aus genau zwei Zeichen besteht. Morse hat Buchstaben als Folge von Punkten und Strichen codiert. Diese Codierungen können in einem Binärbaum dargestellt werden, sodass ein Übergang zum linken Teilbaum einem Punkt und ein Übergang zum rechten Teilbaum einem Strich entspricht. Wenn man im Gesamtbaum startet und durch Übergänge zu linken oder rechten Teilbäumen einen Pfad zum gewünschten Buchstaben sucht, erhält man die Morsecodierung des Buchstabens. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum (Download Q2-I.1)
2. Die Datenstruktur Binärbaum unter Nutzung der Klasse BinaryTree (a) Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen (b) Implementierung der Anwendung oder von Teilen der Anwendung (c) Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf (d) Traversierung eines Binärbaums mit LevelOrder (e) sortiertes Einfügen in Binärbäumen (f) Ermitteln von Höhe und minimaler Blatttiefe eines Binärbaums (g) Verwendung eines Binärbaums im		*Beispiele*: Ahnenbaum (s. o.)

Anwendungskontext		Entropiekodierungen und Huffman-Code
3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree (a) Erarbeitung der Klasse BinarySearchTree unter Benutzung des Interface ComparableContent zur Realisierung einer geeigneten Ordnungsrelation (b) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums		*Beispiele*: gallisches Dorf SuchBaumTestZahlen
4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen		*Beispiele:* Codierungsbäume (s. o.) Huffman-Code Termbaum (s. o.) Bildkomprimierung (QuadTree) Vorschlagslisten in Handys Dateiverwaltungen *oder* *Beispiel*: Buchindex Es soll eine Anwendung entwickelt werden, die anhand von Stichworten und zugehörigen Seitenzahlen ein Stichwortregister erstellt. Da die Stichwörter bei der Analyse des Buches häufig gesucht werden müssen, werden sie in der Klasse Buchindex als Suchbaum (Objekt der Klasse BinarySearchTree) verwaltet. Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle, die nach dem Inhalt

		im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) *oder* *Beispiel*: Entscheidungsbäume (s.o.) *oder* *Beispiel:* Ahnenbaum (s.o.)

Unterrichtsvorhaben Q2-II:

Thema: Endliche Automaten und formale Sprachen

Leitfragen: *Wie kann man (endliche) Automaten genau beschreiben? Wie können endliche Automaten (in alltäglichen Kontexten oder zu informatischen Problemstellungen) modelliert werden? Wie können Sprachen durch Grammatiken beschrieben werden? Welche Zusammenhänge gibt es zwischen formalen Sprachen, endlichen Automaten und regulären Grammatiken?*

Vorhabenbezogene Konkretisierung:
Anhand kontextbezogener Beispiele werden endliche Automaten entwickelt, untersucht und modifiziert. Dabei werden verschiedene Darstellungsformen für endliche Automaten ineinander überführt und die akzeptierten Sprachen endlicher Automaten ermittelt. An einem Beispiel wird ein nichtdeterministischer Akzeptor eingeführt als Alternative gegenüber einem entsprechenden deterministischen Akzeptor.

Anhand kontextbezogener Beispiele werden Grammatiken regulärer Sprachen entwickelt, untersucht und modifiziert. Der Zusammenhang zwischen regulären Grammatiken und endlichen Automaten wird verdeutlicht durch die Entwicklung von allgemeinen Verfahren zur Erstellung einer regulären Grammatik für die Sprache eines gegebenen endlichen Automaten bzw. zur Entwicklung eines endlichen Automaten, der genau die Sprache einer gegebenen regulären Grammatik akzeptiert.

Auch andere Grammatiken werden untersucht, entwickelt oder modifiziert. An einem Beispiel werden die Grenzen endlicher Automaten ausgelotet.

Zeitbedarf: 27 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Endliche Automaten (a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten (b) Untersuchung, Darstellung und Entwicklung endlicher Automaten	Die Schülerinnen und Schüler • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), • analysieren und erläutern Grammatiken regulärer Sprachen (A), • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A), • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), • modifizieren Grammatiken regulärer Sprachen (M), • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M), • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), • ermitteln die Sprache, die ein endlicher Automat akzeptiert (D). • beschreiben an Beispielen den Zusammen-	*Beispiele:* Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen (Download Q2-II.1)
2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen (a) Erarbeitung der formalen Darstellung regulärer Grammatiken (b) Untersuchung, Modifikation und Entwicklung von Grammatiken (c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen, die durch Grammatiken gegeben werden (d) Entwicklung regulärer Grammatiken zu endlichen Automaten (e) (fakultativ) reguläre Ausdrücke formaler Sprachen (f) (fakultativ) Parser formaler Sprachen (g) (fakultativ) Automatenbegriffe und -varianten		*Beispiele:* reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik, Passwort- oder Türcodeprüfmaschine, E-Mail-Adressen-Prüfmaschine, industrielle Fütterungsanlagen, spielerische Sprachen (die himmlische Sprache *Frohlocken*, Dominospiel-Sprache, …), Sporttrainingspläne, Warnstufenanlage für Waldbrandgefahren *Materialien: (s.o.)* *Beispiele*: Moore-Automat, Mealy-Automat

	hang zwischen Automaten und Grammatiken (D).	
3. ggf. Erweiterungen des bisherigen Automatenmodells (a) Kellerautomaten (b) Turingmaschinen (c) Compilerbau		

Unterrichtsvorhaben Q2-III:

Thema: Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Leitfragen: *Was sind die strukturellen Hauptbestandteile eines Computers und wie kann man sich die Ausführung eines maschinenahen Programms mit diesen Komponenten vorstellen? Welche Möglichkeiten bieten Informatiksysteme und wo liegen ihre Grenzen?*

Vorhabenbezogene Konkretisierung:
Anhand einer von-Neumann-Architektur und einem maschinennahen Programm wird die prinzipielle Arbeitsweise von Computern verdeutlicht.

Ausgehend von den prinzipiellen Grenzen endlicher Automaten liegt die Frage nach den Grenzen von Computern bzw. nach Grenzen der Automatisierbarkeit nahe. Mit Hilfe einer entsprechenden Java-Methode wird plausibel, dass es unmöglich ist, ein Informatiksystem zu entwickeln, dass für jedes beliebige Computerprogramm und jede beliebige Eingabe entscheidet, ob das Programm mit der Eingabe terminiert oder nicht (Halteproblem). Anschließend werden Vor- und Nachteile der Grenzen der Automatisierbarkeit angesprochen und der Einsatz von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen beurteilt.

Im Rahmen einer Fall- und Konfliktanalyse werden abschließend ethische Probleme der Informatik angesprochen.

Zeitbedarf: 12 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher b) einige maschinennahe Befehle und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann c) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms	Die Schülerinnen und Schüler • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A), • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).	*Beispiel:* Addition von 4 zu einer eingegeben Zahl mit einem Rechnermodell *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 –Von-Neumann-Architektur und maschinennahe Programmierung (Download Q2-III.1)
2. Grenzen der Automatisierbarkeit a) Vorstellung des Halteproblems b) Unlösbarkeit des Halteproblems c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen		*Beispiel:* Halteproblem *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 - Halteproblem (Download Q2-III.2)
3. Ethische Probleme Fallbeispiele		

Unterrichtsvorhaben Q2-IV:

Wiederholung und Vertiefung ausgewählter Kompetenzen und zurückliegender Inhalte

2.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

In Absprache mit der Lehrerkonferenz sowie unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik des Werner-Heisenberg-Gymnasiums die folgenden fachmethodischen und fachdidaktischen Grundsätze beschlossen. In diesem Zusammenhang beziehen sich die Grundsätze 1 bis 14 auf fächerübergreifende Aspekte, die auch Gegenstand der Qualitätsanalyse sind, die Grundsätze 15 bis 21 sind fachspezifisch angelegt.

Überfachliche Grundsätze:

1) Geeignete Problemstellungen zeichnen die Ziele des Unterrichts vor und bestimmen die Struktur der Lernprozesse.
2) Inhalt und Anforderungsniveau des Unterrichts entsprechen dem Leistungsvermögen der Schüler/innen.
3) Die Unterrichtsgestaltung ist auf die Ziele und Inhalte abgestimmt.
4) Medien und Arbeitsmittel sind schülernah gewählt.
5) Die Schüler/innen erreichen einen Lernzuwachs.
6) Der Unterricht fördert eine aktive Teilnahme der Schüler/innen.
7) Der Unterricht fördert die Zusammenarbeit zwischen den Schülern/innen und bietet ihnen Möglichkeiten zu eigenen Lösungen.
8) Der Unterricht berücksichtigt die individuellen Lernwege der einzelnen Schüler/innen.
9) Die Schüler/innen erhalten Gelegenheit zu selbstständiger Arbeit und werden dabei unterstützt.
10) Der Unterricht fördert strukturierte und funktionale Partner- bzw. Gruppenarbeit.
11) Der Unterricht fördert strukturierte und funktionale Arbeit im Plenum.
12) Die Lernumgebung ist vorbereitet; der Ordnungsrahmen wird eingehalten.
13) Die Lehr- und Lernzeit wird intensiv für Unterrichtszwecke genutzt.
14) Es herrscht ein positives pädagogisches Klima im Unterricht.

Fachliche Grundsätze:

15) Der Unterricht unterliegt der Wissenschaftsorientierung und ist dementsprechend eng verzahnt mit seiner Bezugswissenschaft.
16) Der Unterricht ist problemorientiert und soll von realen Problemen ausgehen und sich auf solche rückbeziehen.
17) Der Unterricht folgt dem Prinzip der Exemplarizität und soll ermöglichen, informatische Strukturen und Gesetzmäßigkeiten in den ausgewählten Problemen und Projekten zu erkennen.
18) Der Unterricht ist anschaulich sowie gegenwarts- und zukunftsorientiert und gewinnt dadurch für die Schülerinnen und Schüler an Bedeutsamkeit.
19) Der Unterricht ist handlungsorientiert, d.h. projekt- und produktorientiert angelegt.
20) Im Unterricht werden sowohl für die Schule didaktisch reduzierte als auch reale Informatiksysteme aus der Wissenschafts-, Berufs- und Lebenswelt eingesetzt.
21) Der Unterricht beinhaltet reale Begegnung mit Informatiksystemen.

2.3 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

Auf der Grundlage von §13 - §16 der APO-GOSt sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe hat die Fachkonferenz des Werner-Heisenberg-Gymnasiums im Einklang mit dem entsprechenden schulbezogenen Konzept die nachfolgenden Grundsätze zur Leistungsbewertung und Leistungsrückmeldung beschlossen. Die nachfolgenden Absprachen stellen die Minimalanforderungen an das lerngruppenübergreifende gemeinsame Handeln der Fachgruppenmitglieder dar. Bezogen auf die einzelne Lerngruppe kommen ergänzend weitere der in den Folgeabschnitten genannten Instrumente der Leistungsüberprüfung zum Einsatz.

2.3.1 Beurteilungsbereich Klausuren

Verbindliche Absprachen:
Bei der Formulierung von Aufgaben werden die für die Abiturprüfungen geltenden Operatoren des Faches Informatik schrittweise eingeführt, erläutert und dann im Rahmen der Aufgabenstellungen für die Klausuren benutzt.

Instrumente:

- Einführungsphase: 1 Klausur je Halbjahr
  Dauer der Klausur: 2 Unterrichtsstunden
- Grundkurse Q 1: 2 Klausuren je Halbjahr
  Dauer der Klausuren: 3 Unterrichtsstunden
- Grundkurse Q 2.1: 2 Klausuren
  Dauer der Klausuren: 4 Unterrichtsstunden
- Grundkurse Q 2.2: 1 Klausur unter Abiturbedingungen
- Anstelle einer Klausur kann in Q 1.2 eine Facharbeit geschrieben werden. Hierzu liegt ein verbindliches Bewertungsraster für das Fach Informatik vor.

Die Aufgabentypen sowie die Anforderungsbereiche I-III sind entsprechend den Vorgaben in Kapitel 3 des Kernlehrplans zu beachten.

Zu Beginn der Q1 werden die Schülerinnen und Schüler über die Möglichkeit, die Rahmenbedingungen und die Bewertungskriterien aufgeklärt, eine Facharbeit im Fach Informatik anzufertigen.

Kriterien

Die Bewertung der schriftlichen Leistungen in Klausuren erfolgt über Hilfspunkte, die den einzelnen Bewertungskriterien zugeordnet sind.

Die Note ausreichend (5 Punkte) soll bei Erreichen von 45 % der Hilfspunkte erteilt werden.

2.3.2 Beurteilungsbereich Sonstige Mitarbeit

Den Schülerinnen und Schülern werden die Kriterien zum Beurteilungsbereich „sonstige Mitarbeit“ zu Beginn des Schuljahres genannt.

Leistungsaspekte

Mündliche Leistungen
- Beteiligung am Unterrichtsgespräch
- Zusammenfassungen zur Vor- und Nachbereitung des Unterrichts
- Präsentation von Arbeitsergebnissen
- Referate
- Mitarbeit in Partner-/Gruppenarbeitsphasen
- Problem- und Frage-Formulierungen

Praktische Leistungen am Computer
- Implementierung, Test und Anwendung von Informatiksystemen

Sonstige schriftliche Leistungen
- Lernerfolgsüberprüfung durch kurze schriftliche Übungen
  Über die Durchführung schriftlicher Übungen entscheidet je nach Verlauf der Mitarbeit in den Kursen die jeweilige Lehrkraft.
  Schriftliche Übung dauern nicht länger als 30 Minuten und umfassen den Stoff der letzten 4–6 Stunden.
- Bearbeitung von schriftlichen Aufgaben im Unterricht

Kriterien

Die folgenden allgemeinen Kriterien gelten sowohl für die mündlichen als auch für die schriftlichen Formen der sonstigen Mitarbeit.

Die Bewertungskriterien stützen sich auf
- die Qualität der Beiträge,
- die Quantität der Beiträge und
- die Kontinuität der Beiträge.

Besonderes Augenmerk ist dabei auf
- die sachliche Richtigkeit,
- die angemessene Verwendung der Fachsprache,
- die Darstellungskompetenz,
- die Komplexität und den Grad der Abstraktion,
- die Selbstständigkeit im Arbeitsprozess,
- die Präzision und
- die Differenziertheit der Reflexion zu legen.

Bei Gruppenarbeiten auch auf
- das Einbringen in die Arbeit der Gruppe,
- die Durchführung fachlicher Arbeitsanteile und
- die Qualität des entwickelten Produktes.

Bei Projektarbeit darüber hinaus auf

- die Dokumentation des Arbeitsprozesses,
- den Grad der Selbstständigkeit,
- die Reflexion des eigenen Handelns und
- die Aufnahme von Beratung durch die Lehrkraft.

Grundsätze der Leistungsrückmeldung und Beratung

Die Grundsätze der Leistungsbewertung werden zu Beginn den Schülerinnen und Schülern transparent gemacht. Leistungsrückmeldungen können erfolgen

- nach einer mündlichen Überprüfung,
- bei Rückgabe von schriftlichen Leistungsüberprüfungen,
- nach Abschluss eines Projektes,
- nach einem Vortrag oder einer Präsentation,
- bei auffälligen Leistungsveränderungen,
- auf Anfrage,
- als Quartalsfeedback und
- zu Elternsprechtagen.

Die Leistungsrückmeldung kann

- durch ein Gespräch mit der Schülerin oder dem Schüler,
- durch einen Feedbackbogen,
- durch die schriftliche Begründung einer Note oder
- durch eine individuelle Lern-/Förderempfehlung

erfolgen.

2.4 Lehr- und Lernmittel

Da nach Sichtung der wenigen auf dem Markt befindlichen Lehrwerke für den Informatikunterricht in der Sekundarstufe II festgestellt wurde, dass kein Werk *alle* relevanten Themen bedient und die in ihnen angeleiteten Projekte nicht auf Zustimmung der Fachgruppe stoßen, wurde entschieden, kein verbindliches Lehrwerk festzulegen. Zwecks zügiger Umsetzung von Projekten und Informationsweitergabe stellen die Lehrkräfte individualisierte Java-Vorgabedateien sowie Arbeitsblätter in digitaler Form zur Verfügung.

3 Entscheidungen zu fach- und unterrichtsübergreifenden Fragen

Bezüge zu anderen Fächern
Im Informatikunterricht werden Kompetenzen anhand informatischer Inhalte in verschiedenen Anwendungskontexten erworben, in denen Schülerinnen und Schülern aus anderen Fächern Kenntnisse mitbringen können. Diese können insbesondere bei der Auswahl und Bearbeitung von Softwareprojekten berücksichtigt werden und in einem hinsichtlich der informatischen Problemstellung angemessenem Maß in den Unterricht Eingang finden. Insbesondere finden sowohl inhaltliche Grundvoraussetzungen aus den Fächern Mathematik und Physik (vor allem in den Inhaltsfeldern *Informatiksysteme*, *Algorithmen* und *formale Sprachen und Automaten*, aber auch rein mathematisch-inhaltliche Fertigkeiten etwa zur Bruchrechnung oder zu einfachen trigonometrischen Sachverhalten) als auch fachliche Querverweise zu den Fächern Geschichte, Sozialwissenschaften und Philosophie (vor allem im Inhaltsfeld *Informatik, Mensch und Gesellschaft*, in dem wichtige historische, aktuelle gesellschaftliche und ethische Fragen erörtert werden) ihren angemessenen Platz im Informatikunterricht.

Vorbereitung auf die Erstellung der Facharbeit
Möglichst schon im zweiten Halbjahr der Einführungsphase, spätestens jedoch im ersten Halbjahr des ersten Jahres der Qualifikationsphase werden im Unterricht an geeigneten Stellen Hinweise zur Erstellung von Facharbeiten gegeben. Das betrifft u. a. Themenvorschläge, Hinweise zu den Anforderungen und zur Bewertung. Es wird vereinbart, dass nur Facharbeiten vergeben werden, die mit der eigenständigen Entwicklung eines Softwareproduktes verbunden sind. Eine entsprechende Eigeninitiative bei der Themenauswahl und Ausgestaltung der Arbeitsaufträge wird von den Schülerinnen und Schülern erwartet, gestützt von der Beratung des Fachlehrers.

4 Qualitätssicherung und Evaluation

Das schulinterne Curriculum (siehe 2.1) wird im Rahmen der regelmäßig stattfindenden Fachkonferenzen Informatik auf der Grundlage der Unterrichtserfahrungen überprüft und bei Bedarf angepasst.